Domensko specifični jezik za opis vsega, kar je povazno s kolesarstvom v Mariboru

Predmet: Prevajanje programskih jezikov

Skupina: OnlyFriends

Člani: Marko Roškar, Jakob Oprešnik, Erik Lašič

Konstrukti

Osnovni konstrukti:

števila, nizi, točke

Bloki:

posebne točke (parkirna mesta, kolesarnice, mBajk izposojevalnice),

kolesarska pot, turistična kolesarska pot, koridor,

zgradba, reka, park

Ukazi:

line, bend, box

Dodatni konstrukti

Spremenljivke in konstante, izrazi:

```
const z = 4
var i = 5 + z
```

Zanke in vejitve:

```
If (1 > 2) {
   line((1, 2), (2, 4), 3)
} elseif (2 == 3) {
   line((3, 3), (5, 5), 3)
} else {
   line((4, 4), (6, 6), 3)
}
```

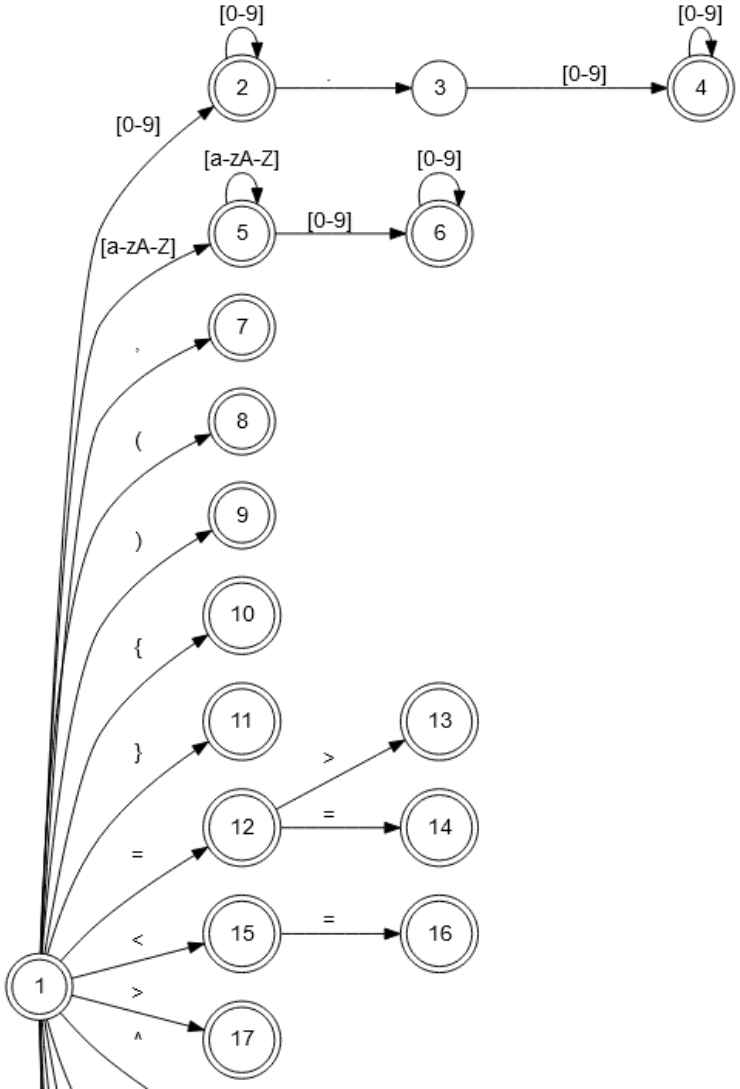
Abstrakcije:

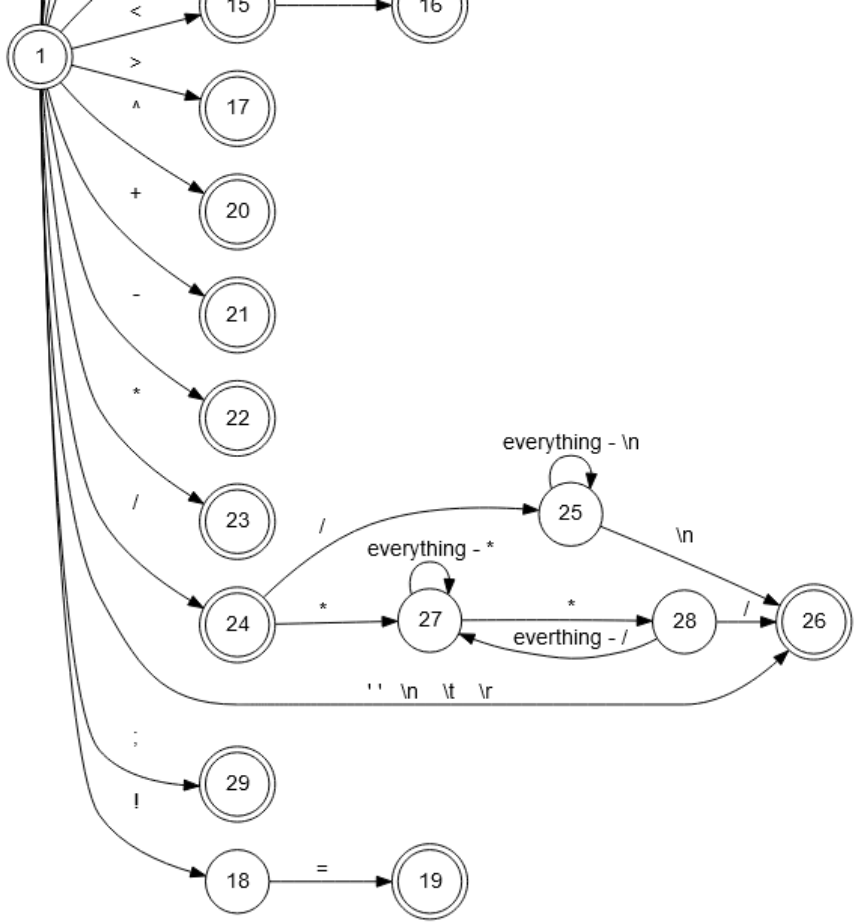
```
func sum(x, y) {
   var sum = 0
   sum = x + y
   return sum
}

var z = call sum(447575.6057786940, 100000.0000000002)
```

Komentarji:

```
/* ... */

//
```

Avtomat (graphviz)





BNF + ANTLR (sintaksa)

```
PROGRAM ::= FUNCTIONS CITY
FUNCTIONS ::= FUNCTION FUNCTIONS | epsilon
CITY ::= city name { BLOCKS }
BLOCKS ::= BLOCK MOREBLOCKS
MOREBLOCKS ::= BLOCK MOREBLOCKS | epsilon

BLOCK ::= ROAD | BIKEPATH | TOURPATH | CORRIDOR | BUILDING | RIVER | PARK | MARKPOINT
ROAD ::= road name { COMMANDS }
BIKEPATH ::= bikePath name { COMMANDS }
TOURPATH ::= bikeTourPath name { COMMANDS }
CORRIDOR ::= bikeCorridor name { COMMANDS }
BUILDING ::= building name { COMMANDS }
RIVER ::= river name { COMMANDS }
PARK ::= park name { COMMANDS }

MARKPOINT ::= BIKESTAND | BIKESHED | MBAJK | RENTBIKE
BIKESTAND ::= bikeStand(integer) name { POINT }
BIKESHED ::= bikeShed name { POINT }
MBAJK ::= mBajk name { POINT }
RENTBIKE ::= rentBike name { POINT }


COMMANDS ::= COMMAND MORECOMMANDS
MORECOMMANDS ::= COMMAND MORECOMMANDS | epsilon
COMMAND ::= LINE | BEND | BOX | FOR | IF | VARIABLE | CONST | ASSIGNMENT
LINE ::= line( POINTORVAR , POINTORVAR , WIDTH )
BEND ::= bend( POINTORVAR , POINTORVAR , ANGLE )
ANGLE ::= integer | variable_name    //integer ::= [0-9]+
WIDTH ::= integer | variable_name
BOX ::= box( POINTORVAR , POINTORVAR )
POINTORVAR ::= POINT | identifier
```

```
grammar antlr;

program: func* city EOF;
city: 'city ' name '{' blocks '}';

blocks: block+;
block: road | bikepath | tourpath | coridor | building | river | park | marked;

road: 'road ' name '{' commands '}';

bikepath: 'bikePath ' name '{' commands '}';

tourpath: 'bikeTourPath ' name '{' commands '}';

coridor: 'bikeCoridor ' name '{' commands '}';

building: 'building ' name '{' commands '}';

river: 'river ' name '{' commands '}';

park: 'park ' name '{' commands '}';

marked: bikestand | bikeshed | mbajk;
bikestand: 'bikeStand ' name point;
bikeshed: 'bikeShed ' name point;
mbajk: 'mBajk ' name point;
```

Izračun First & Follow

```
FIRST:
    F : { (, float, identifier, integer }
    Y : { -, +, (, float, identifier, integer }
    XX : { ^, epsilon }
    X : { -, +, float, identifier, integer }
    TT : { *, /, epsilon }
    T : { -, +, float, identifier, integer }
    EE : { +, -, epsilon }
    E : { -, +, float, identifier, integer }
    EXPR : { -, +, float, identifier, integer }
    RETURNVALUE : { (, float, identifier, integer }
    RETURN : { return }
    MOREVARS : { ',' , epsilon }
    ARGS : { identifier, epsilon }
    FUNCTION : { func }
    VARIABLENAMEORNUBMER : { identifier, integer, float }
    MOREVORN : { ',', epsilon }
    CALLARGS : { identifier, integer, float }
    CALL : { call }
    VALUE : { float, integer, identifer, (, call }
    ASSIGNMENT : { identifier }
    CONST : { const }
    VARIABLE : { var }
    POINT : { ( }
```

```
FOLLOW:
    PROGRAM : { EOF }
    FUNCTIONS : { city }
    CITY : { EOF }
    BLOCKS : { '}' }
    MORE_BLOCKS : { '}' }
    BLOCK : FIRST(MOREBLOCKS) + FOLLOW(MOREBLOCKS) U FIRST(MOREBLOCKS) + FOLLOW(
    = { road, bikePath, tourPath, corridor, building, river, park, rentBike, bik
    + { road, bikePath, tourPath, corridor, building, river, park, rentBike, bik
    = { road, bikePath, tourPath, corridor, building, river, park, rentBike, bik
    ROAD : { road, bikePath, tourPath, corridor, building, river, park, rentBike
    BIKEPATH : { road, bikePath, tourPath, corridor, building, river, park, rentB
    TOURPATH : { road, bikePath, tourPath, corridor, building, river, park, rentB
    CORRIDOR : { road, bikePath, tourPath, corridor, building, river, park, rentB
    BUILDING : { road, bikePath, tourPath, corridor, building, river, park, rentB
    RIVER : { road, bikePath, tourPath, corridor, building, river, park, rentBik
    PARK : { road, bikePath, tourPath, corridor, building, river, park, rentBike
    MARKPOINT : { road, bikePath, tourPath, corridor, building, river, park, rent
    BIKESTAND : { road, bikePath, tourPath, corridor, building, river, park, rent
    BIKESHED : { road, bikePath, tourPath, corridor, building, river, park, rentB
    MBAJK : { road, bikePath, tourPath, corridor, building, river, park, rentBik
    RENTBIKE : { road, bikePath, tourPath, corridor, building, river, park, rentB
    COMMANDS : { '}', return }
    MORECOMMANDS : { '}', return}
    COMMAND : { line, bend, box, for, if, var, const, identifier, '}', return }
    LINE : { line, bend, box, for, if, var, const, identifier, '}', return }
    BEND : { line, bend, box, for, if, var, const, identifier, '}', return }
```

Implementacija jezika

```
class MBajk(private val name: String, private val point: Point, override var next: Stmt): Stmt{
    override fun toString(): String{
        return "mBajk $name { $point }\n$next"
    }

    override fun evalPartial(env: Map<String, Expr>): Stmt {
        currentBlock="mBajk"
        return MBajk(name, point.evalPartial(env) as Point, next.evalPartial(mutableMapOf()))
    }

    override fun toGeoJSON(): String {
        currentBlock="mBajk"
        return  "    {\n" +
                "      \"type\": \"Feature\",\n" +
                "      \"properties\": {\n" +
                "        \"marker-color\": \"${blockColor[currentBlock]}\",\n" +
                "        \"class\": \"${currentBlock}\"\n"+
                "      },\n" +
                "      \"geometry\": {\n" +
                "        \"type\": \"Point\",\n" +
                "        \"coordinates\": [\n" +
                "          ${point.longitude},\n" +
                "          ${point.latitude}\n" +
                "        ]\n" +
                "      }\n" +
                "    },\n${next.toGeoJSON()}"
    }
}
```

```
private fun parseMBAJK(): MBajk {
    parseTerminal(M_BAJK)
    val mBajkName=parseTerminal(IDENTIFIER)
    parseTerminal(LB_PAREN)
    val point=parsePOINT()
    parseTerminal(RB_PAREN)
    return MBajk(mBajkName, point, End)
}
```

Zanimivosti

```
val RESERVED_WORDS = mapOf(
    "bend" to BEND, "bikeCorridor" to BIKE_CORRIDOR, "bikePath" to BIKE_PATH, "bikeShed" to BIKE_SHED,
    "bikeStand" to BIKE_STAND, "bikeTourPath" to BIKE_TOUR_PATH, "box" to BOX, "building" to BUILDING, "call" to CALL,
    "const" to CONST, "circ" to CIRC, "city" to CITY, "else" to ELSE, "elseif" to ELSE_IF, "for" to FOR, "func" to FUNC,
    "if" to IF, "line" to LINE, "mBajk" to M_BAJK, "park" to PARK, "rentBike" to RENT_BIKE, "return" to RETURN,
    "river" to RIVER, "road" to ROAD, "to" to TO, "var" to VAR
)

class Assignment(private val name: String, private val value: Expr, override var next: Stmt): Stmt{
    override fun toString(): String{
        return "\t$name=$value\n$next"
    }

    override fun evalPartial(env: Map<String, Expr>): Stmt {
        if(env.contains(name)){
            return next.evalPartial( env: env + (name to value.evalPartial(env)))
        }else if(env.contains("-const$name")){
            throw CantAssignException(name)
        }else {
            throw UndefinedException(name)
        }
    }
}
```

Primer programa

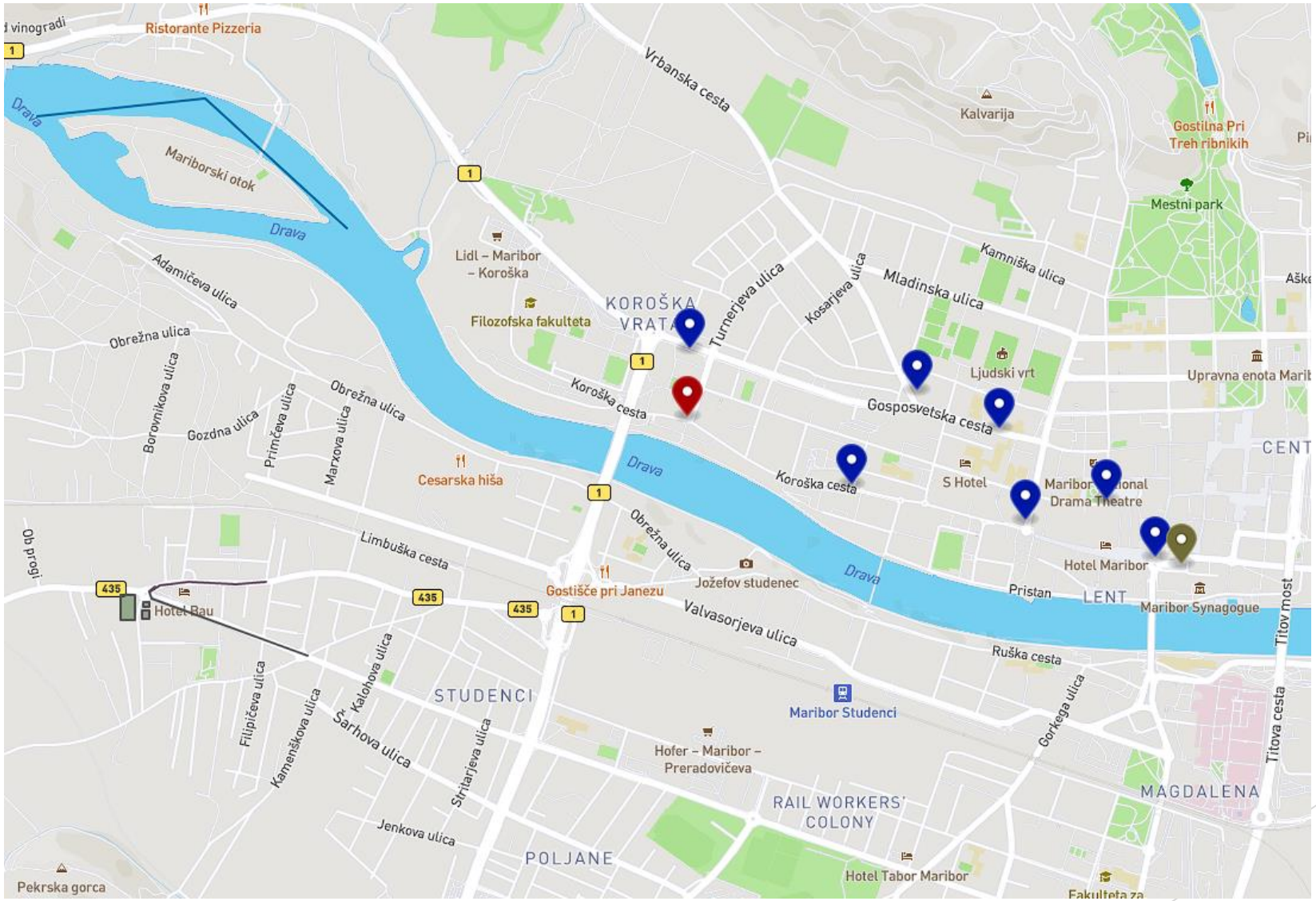
```
func sum(x, y){
    var sum=0
    sum = x+y
    return sum
}

city Maribor{
    mBajk mBike1{(547575.6292748451, 46.56272584156842)}
    mBajk mbike2{(547575.6413984299, 46.55847668504357)}
    mBajk mBike3{(547575.6374716759, 46.56169309108066)}
    mBajk mBike4{(547575.6351327896, 46.55935088766312)}
    mBajk mBike5{(547575.6404650211, 46.56075253335337)}
    mBajk mBike6{(547575.6443059444, 46.55898571612989)}
    mBajk mBike7{(547575.6460762024, 46.55757295866902)}
    bikeShed bikeShed1{(547575.6470364332, 46.55738483414326)}
    bikeStand(5) bikeStand1{(547575.6292104721, 46.56103654662082)}
    building building1{box((547575.6095901132, 46.556309560855844), (547575.6098207831, 46.556451579303726))}
    building building2{box((547575.6095445156, 46.55604212250993), (547575.6098020077, 46.55622287408808))}
    //komentar
    /*
    Več vrstični komentar
    */
    park park1{box((547575.6087934971, 46.555986790273806), (547575.6092816591, 46.55660650809569))}
    road road1{
        line((547575.6155204773, 46.55511253345494), (547575.6098878384, 46.55654379887603), 2)
        line((547575.6098878384, 46.55654379887603), (547575.6098221242, 46.55670518275048), 2)
        line((547575.6098221242, 46.55670518275048), (547575.6100460886, 46.55682414541347), 2)
    }
```

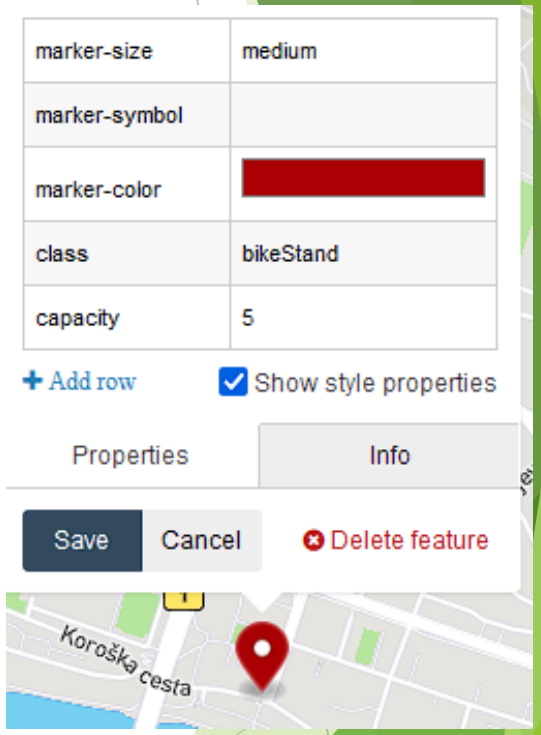
```
river river1{
    var z=call sum(447575.6057786940, 100000.0000000002)
    bend((z, 46.56847937726347), (547575.6169366837, 46.56570595418568), 35)
    bend((547516.40625, 1.9332268264771233), (547558.9453125, 69.16255790810501), 105)
    /*
    if(1>2){
        line((1,2),(2,4), 3)
    }elseif(2==3){
        line((3,3),(5,5), 3)
    }else{
        line((4,4),(6,6), 3)
    }
    */
}
```

Izhod programa


```
{
  "type": "FeatureCollection",
  "features": [
       {
      "type": "Feature",
      "properties": {
        "marker-color": "#0317a7",
        "class": "mBajk"
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          547575.6292748451,
          46.56272584156842
        ]
      }
    },
    {
      "type": "Feature",
      "properties": {
        "marker-color": "#0317a7",
```

Prikaz v orodju geojson.io





HVALA ZA POZORNOST